NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

REINFORCEMENT LEARNING: A NEW APPROACH FOR THE CULTURAL GEOGRAPHY MODEL

by

Sotirios Papadopoulos

September 2010

Thesis Advisor:	Christian J. Darken
Second Reader:	Jonathan K. Alt

This thesis was done at the MOVES Institute
Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 2010	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Reinforcement Learning: A New Approach for the Cultural Geography Model	5. FUNDING NUMBERS
6. AUTHOR(S) Sotirios Papadopoulos	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ___N.A.___.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

The Cultural Geography (CG) model, under development in TRAC Monterey, is an open-source agent-based social simulation, designed to offer an insight into the response of the civilian population during Irregular Warfare (IW) operations. It implements social and behavioral science theories that govern the behaviors of agents within the simulation using Bayesian belief networks.

At this stage, the agents within the CG model do not select their actions at all. Instead, all their actions are hard coded into the model's scenario file. As part of an attempt to improve the model, this effort sought to enhance the functionality within the model by exploring the use of utility functions and, more specifically, the concept of reinforcement learning.

This study began with the development of a learning agent prototype. After the initial testing for its functionality, the code that was developed was inserted into the main CG model. Based on specially developed scenarios, and by employing a design of experiments methodology, we created experimental runs. By applying statistical and analysis techniques, we showed that reinforcement learning works properly inside the Social Network environment and produces the desired results.

This study can be used as a starting point for the research of the effects of reinforcement learning in social modeling in general.

14. SUBJECT TERMS Cultural Geography, Social Simulations, Reinforcement Learning, Agent-Based Modeling, Irregular Warfare (IW)	15. NUMBER OF PAGES 71
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

REINFORCEMENT LEARNING: A NEW APPROACH FOR THE CULTURAL GEOGRAPHY MODEL

Sotirios Papadopoulos
Lieutenant Colonel, Hellenic Army
Hellenic Army Academy, June 1989

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MODELING, VIRTUAL ENVIRONMENTS AND SIMULATION (MOVES)

from the

NAVAL POSTGRADUATE SCHOOL
September 2010

Author: Sotirios Papadopoulos

Approved by: Christian J. Darken
Thesis Advisor

Jonathan K. Alt
Second Reader

Mathias Kolsch
Chair, MOVES, Academic Committee

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Cultural Geography (CG) model, under development in TRAC Monterey, is an open-source agent-based social simulation, designed to offer an insight into the response of the civilian population during Irregular Warfare (IW) operations. It implements social and behavioral science theories that govern the behaviors of agents within the simulation using Bayesian belief networks.

At this stage, the agents within the CG model do not select their actions at all. Instead, all their actions are hard coded into the model's scenario file. As part of an attempt to improve the model, this effort sought to enhance the functionality within the model by exploring the use of utility functions and, more specifically, the concept of reinforcement learning.

This study began with the development of a learning agent prototype. After the initial testing for its functionality, the code that was developed was inserted into the main CG model. Based on specially developed scenarios, and by employing a design of experiments methodology, we created experimental runs. By applying statistical and analysis techniques, we showed that reinforcement learning works properly inside the Social Network environment and produces the desired results.

This study can be used as a starting point for the research of the effects of reinforcement learning in social modeling in general.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ANSF	Afghanistan National Security Forces
CG	Cultural Geography
COA	Course of Action
COIN	Counterinsurgency
DOE	Design of Experiments
GIRoA	Government of the Islamic Republic of Afghanistan
IED	Improvised Explosive Device
ISAF	International Security Afghanistan Forces
IW	Irregular Warfare
MOE	Measure of Effectiveness
NGOs	Non-Governmental Organizations
NOLH	Nearly Orthogonal Latin Hypercube
TRAC	TRADOC Analysis Center
TRADOC	Training and Doctrine

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

During the course of this study, I had the privilege of coming into contact and receiving support from many people. Without their support, the study would not have the form it now does. A few words about each one of them are the least I can offer.

First, I would like to thank Dr. Chris Darken, my thesis advisor, for the inspiration. His teachings were the foundation for my research and his continuous suggestions gave to this study its present form.

Lieutenant Colonel Jon Alt was my “other half” for the duration of this study. All my thoughts and actions were filtered by him. His enthusiasm and encouragement were always a source of strength for me. His passion for the science of Social Modeling knows no bounds. I am certain that his future dissertation work will make people take notice.

TRAC Monterey provided support to me throughout the course of this study, but I should give special mention to two people who stood above the rest. Major Francisco Baez, a tireless TRAC analyst, assisted greatly in the construction of the model scenarios for this study. Mr. Harold Yamauchi, a programming mastermind, got me out of dark programming places throughout this study and incorporated my code into the main Cultural Geography model with great success.

A thesis project requires many hours spent in front of a computer screen. When you have a family of five, these hours can be hard to find. I was lucky enough to have a person within my family who made sure I had the necessary conditions to work in by single-handedly taking care of the home front. If not for this person, the work you are about to read would never be completed on time. My eternal gratitude goes to this person, my wife, Vanessa.

THIS PAGE INTENTIONALLY LEFT BLANK

I. A NEW APPROACH

A. INTRODUCTION

Irregular Warfare (IW) is defined as “a violent struggle among state and non-state actors for legitimacy and influence over the relevant population” (JP 3-0, Chapter I, p. 6).

It is obvious, by reading the above definition, that the main focus of IW is the population. As a response to this fact, the need arose for a model that could represent the target population in an adequate and realistic way. The Cultural Geography (CG) model, under development in TRAC Monterey, is an open-source agent-based social simulation, designed to offer insight into the response of the civilian population during Irregular Warfare (IW) operations. It implements social and behavioral science theories that govern the behaviors of agents within the simulation using Bayesian belief networks.

B. PROBLEM STATEMENT

At this stage, the agents within the model do not select their actions at all. Instead, all their actions are hard coded into the model’s scenario file. Although this approach allows the user to explore the impact of a certain sequence of events on the population that is being studied, it detracts from the realism of the scenario execution and also does not allow the actors within the model to explore, and potentially find, the courses of action that could be more useful for their purposes.

As part of an attempt to improve the model, this effort will seek to enhance the functionality within the model by exploring the use of utility-based action selection, closely related to reinforcement learning.

C. RESEARCH QUESTIONS

The main questions that this study will try to answer are:

- Is reinforcement learning appropriate for use in social simulations and the CG model in particular?
- What are the advantages of using utility-based agents within the CG model?

D. BENEFITS OF THE STUDY

This study is expected to impact directly the functionality of the CG model by enhancing its capabilities and augmenting its amount of realism. The new functionality is expected to give to the user the potential to explore a greater amount of possible action sequences in his attempt to determine the sequence that results in optimal results for his purposes. It will also help improve the insights that the model provides about the target population, and, as a result, arm decision makers with more realistic and rich information about their operational environment.

The scope of his study is not limited to the CG model. Its concepts, and the results that stem from it, should be considered as applicable within the realm of Social Networking in general. Future research could look into applying the concept of reinforcement learning to other Social Models.

E. METHODOLOGY

The study will begin with the development of an agent prototype that will use utility-based reinforcement learning as its driving force. After the initial testing of the functionality of this prototype, the concept will be applied to all agents within the model. A scenario will be designed, with special focus on infrastructure. Based on this scenario, we will design an experiment, varying certain parameters that affect the reinforcement learning process. A statistical analysis of the results will attempt to illustrate the agents' proper functionality and answer the research questions stated above.

F. WHAT COMES NEXT

Chapter II will lay the cognitive foundation for this study by explaining the concept of reinforcement learning. Chapter III will illustrate the process that was followed for the creation of the learning agent prototype. It will also present the results of the initial test run that was performed to test it. Chapter IV will introduce the CG model and its components. It will also describe the experiment that took place to validate the new agent's functionality and the results of the statistical analysis that followed. Finally, in Chapter V, the study is concluded by presenting a brief discussion of the analysis results and by providing suggestions for possible future work on this area of the CG model.

THIS PAGE INTENTIONALLY LEFT BLANK

II. LOOKING INTO THE PAST

A. INTRODUCTION

The purpose of this chapter is to present to the reader a quick overview of the concept of reinforcement learning. This concept is the basis upon which the creation of the utility-based agent for the CG model was based. First, we will begin with an overview, in simple terms, of how reinforcement learning works and, finally, we will finish with an argument about the applicability of reinforcement learning to social modeling, in general, and the CG model, in particular.

B. REINFORCEMENT LEARNING

The term “reinforcement learning” describes a concept in which an agent uses certain techniques to understand his environment, through the percepts that he is receiving from it and the rewards he received from his past actions, and eventually decides on his next course of action in order to maximize his potential future reward, as he estimates it (Russell & Norvig, 2003) Based on this simple explanation of reinforcement learning, we can determine the two main building blocks of this concept: rewards policy and decision process. In the following subsections, I will try to explain how these two blocks work together to produce an intelligent agent.

1. Useful Terms

Before we begin our discussion on rewards policy and decision process, it is deemed necessary to clarify some terms that will be used in this chapter and throughout the study. This will help the reader understand the concept that these terms are being used to describe.

a. Percepts

Every agent uses the mechanisms available to him (sensors, software, etc.) to assess the state of his environment at every moment. The results of this assessment are called percepts. Therefore, in the case of an agent playing chess, a percept could be the current state of the chess board. A set of these percepts, at a given point in time, form the state of the environment the agent exists in. In reinforcement learning, an agent keeps track of the sequence of these states he is in, in order to formulate a better understanding of his environment and develop a strategy. The development of this strategy will be discussed in the next subsection (Russell & Norvig, 2003).

b. Utility

A distinction must be made between “point utility” and “utility per unit time.” We can define “point utility” as the reward that is received at a specific point in time, as a result of an action. “Utility per unit time” is the reward that is received over a unit of time. In this study, whenever we refer to utility, we assume “point utility.”

2. Rewards Policy

A widely accepted hypothesis is that all humans, whether they realize it or not, plan a great part of their actions based on the promise of a reward. This reward can be material (money, promotion, etc.) or immaterial (self happiness, inner peace, etc.). It can be argued that an agent inside a simulation cannot act based on immaterial rewards, since he cannot feel (although some people might disagree with that point, claiming that feelings could somehow be modeled). Therefore, if we accept that an agent cannot feel, the only way we can reward such an agent is through material means. This reward is the result of a utility function, and is represented by a numeric value. A utility function is defined by the agent’s creator and drives the agent’s actions throughout the duration of the

simulation. Each time an agent acts, a reward is given according to the results of the action. The reward acts as a feedback to the agent about the actions he is choosing.

According to Russell and Norvig (2003, p. 51) “a utility function maps a state (or a sequence of states) onto a real number, which describes the associated degree of happiness.” The approach of this study is slightly different. Instead of being attributed to states, the credit for rewards is assigned to actions, thus making the reinforcement learning process independent of the various states the agent might find himself in.

In general, an agent tries to select actions that provide the greatest expected reward based on the discounted reward attributable to that action. To do that, the agent must keep track of all his past actions and the rewards associated with them. Whenever the agent must make a new choice, he must revisit his past actions and see what happened. By doing that, he calculates the expected utility (expected reward) for each of his candidate actions.

To better illustrate this procedure, we will use a simple example, as shown in Figure 1.

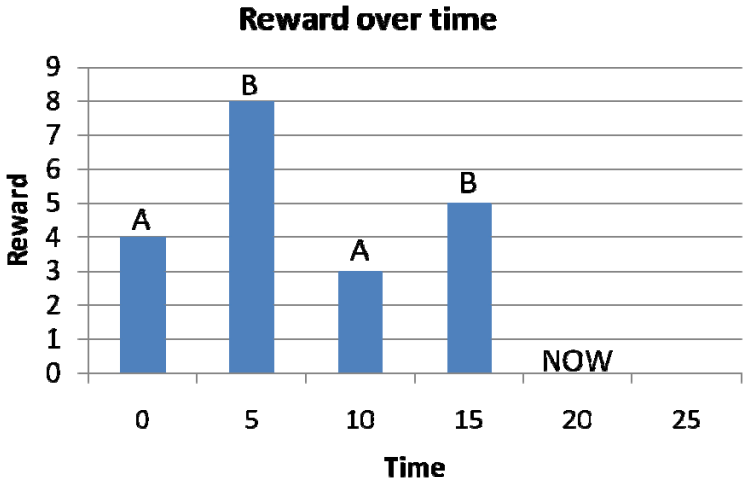


Figure 1. Sample firing of actions and associated rewards

Let us assume that an agent, at the present point in time (time 20), has to choose between two candidate actions, A and B. The agent, as mentioned above, must calculate the expected utility for each of these two actions. Let us start with action A. The agent looks into his past actions (Figure 1) and locates the points in time when action A was executed (fired). Action A was fired two times, at time points 0 and 10. For each of these firings, we calculate the total discounted utility that was received after that specific firing time. The formula that is used for this calculation is:

$$U(t) = \sum_{i=1}^{k} r_i \lambda^{t_i - \tau}$$

In that formula, U is the total utility for the action that was taken at time τ, k is the number of rewards that were awarded after time t_i, r_i is the reward that was awarded at time t_i, and λ is the discount factor. In the case we are examining, for the first firing of action A, the total utility is:

$$U(t_0) = 8 \cdot \lambda^{5-0} + 3 \cdot \lambda^{10-0} + 5 \cdot \lambda^{15-0}$$

We should note that, in the above calculation, we used all utilities that were awarded after time 1, no matter the action with which they were associated.

The same procedure is repeated for the firing of action A that took place at time 10. For that firing, the total utility is:

$$U(t_{10}) = 5 \cdot \lambda^{15-10}$$

The expected utility for an action is given by the following formula:

$$\overline{U} = \frac{1}{n} \sum_{j=1}^{n} U(t_j)$$

In that formula, $\overline{U}$ is the expected utility for the action, and n is the number of firings for that action. In our example, the expected utility for action A is:

$$\overline{U} = \frac{1}{2}\left[U(t_0) + U(t_{10})\right]$$

This discount factor (lambda) is user defined and can take its values between 0 and 1. The meaning of the discount rate is the following: a discount rate closer to 0 drives the agent to maximize his short-term reward by not giving too much value to future rewards. In contrast, a discount rate closer to 1 drives the agent to maximize his long-term reward. When lambda takes the value of 1, the agent considers all his future rewards in an additive way (adds them all as they are, without any discounting).

3. Decision Process

After the calculation of the expected utility for each candidate action, the agent must decide which action he will actually perform. Two simple action selection methods will be discussed below, the ε-greedy action selection method and the softmax action selection method. The information presented below is explained in far more depth by Sutton and Barto (1998, p. 27-31).

a. ε-greedy Method

During this method, the agent selects, most of the time, the action with the highest expected utility, but sometimes, with a user-defined low probability ε, he makes his selection randomly and uniformly among the candidate actions, without caring about the expected utility of these actions. By doing that, the agent is able to explore more fully his candidate actions and perhaps eventually reach a more optimal result. It has been shown (Sutton and

Barto, 1998, p. 28-29) that the ε-greedy method produces better results than the pure greedy method, during which the agent always chooses the action with the highest expected utility.

b. Softmax Method

During this method, all actions are assigned probabilities that correspond to their expected utility. So, the action with the highest expected utility gets the highest probability, etc. These probabilities are assigned by the following formula, which represents the Boltzmann distribution:

$$P_i = \frac{e^{E_i/t}}{\sum_j e^{E_j/t}}$$

In that formula, P_i is the probability assigned to action i, E_i is the expected utility of action i, and t is a parameter called temperature. The temperature takes values that are greater than zero. A higher value of temperature makes the agent more adventurous in his decisions. This means that the agent is more likely to choose an action with a lower value of P_i. As the temperature approaches 0, the agent becomes greedier and chooses the action with the highest value of P_i. It must be noted that this “softmax effect” can be achieved in other ways other than the Boltzmann distribution.

c. Brief Discussion

The main difference between the two methods described above lies in what the agent does when he is not choosing in a greedy way. In the ε-greedy method, the agent chooses randomly and uniformly among the candidate actions. In the softmax method, the agent does not choose in a uniform way, but takes into account the probabilities that are assigned to each action and, by extension, the expected utilities of the actions. It is clear that, to reach an optimal

result, the agent must balance exploration and exploitation in his action selection. The level of this balancing cannot be predetermined, since it is closely connected to the nature of the tasks that an agent must perform. In this study, we will use the softmax method for action selection, and the probabilities for the candidate actions will be calculated by using the Boltzmann distribution formula, as described above.

4. Other Considerations

The application of reinforcement learning shows potential to impact the following areas of social simulation:

- **Realism**: the agents behave in a way closer to the way an actual human might behave
- **Flexibility**: through a small number of parameters, the user can change the way the agents behave and, by doing so, explore an infinite number of action sequences.
- **Increased capabilities**: Observational data shows that the scenario execution time becomes considerably lower, when compared to the time it takes to run a scenario with hard coded actions, thus allowing the users to use more agents in their scenarios and model more complex situations.
- **Traceability for analysis**: the users can collect data about preferred action choices by the agents and provide an analysis of the potential reasons and motives behind those choices.

In the case of the CG model, the transition from hard-coded actions to reinforcement learning enhanced the model's capabilities without compromising any of the functionalities that the model provided in the past. The analysis that will be provided in Chapter IV can be considered as a verification step for the integration of utility-based action-selection code into the CG model.

THIS PAGE INTENTIONALLY LEFT BLANK

III. THE CREATION OF THE UTILITY-BASED AGENT

A. INTRODUCTION

The purpose of this chapter is to give a detailed overview of the procedure that was followed for the creation of the utility-based agent inside the Cultural Geography (CG) model. We will show how the concept of reinforcement learning, as detailed in Chapter II, was implemented into the CG model code. The information presented in this chapter is intended to help the reader understand the reasoning behind the creation of the various new components for the CG model. To fully understand the code that was created, the reader must also be fairly familiar with Java programming.

B. THE BASICS

A utility-based agent is an agent who can decide on its actions based on a certain procedure. The introduction of such an agent in the CG model is a revolutionary move since, thus far, the actions of all agents inside the model were hard coded inside the Excel configuration file. For the creation of our agent, we based our design on the utility theory. We utilized a utility discount method and the final choosing of the action was based on the Boltzmann distribution. The theoretical background behind these techniques was covered in Chapter II, while the application of these techniques will be described in more detail in subsequent sections of this chapter.

We decided to build our template for application on the insurgents component of the CG model. By doing that, we intended to replace any hard-coded insurgent actions from the scenario. From that point on, the insurgents would perform actions (or do nothing) according to our utility-based procedure. Initially, we created one such agent for the whole model.

Since we decided to use the insurgents component as our “guinea pig,” we needed to define a utility in accordance with the insurgents’ needs and interests. We concluded that the average change in the population’s stance on the issue of security would be a good enough measure of utility for our agent. Since our agent, being an insurgent, would like to decrease the notion of security inside the population, it makes sense to assume that a decrease in the population’s stance on security should result in a positive utility value for our agent.

Five new classes were created in the CG model to support the new agent:

ActionEnergy – This class is used to create ActionEnergy objects. These objects are utilized to store the expected utility values for each agent and each action. The list is updated with each activation of the utility agent.

AgentAction – This class is used to create AgentAction objects. These objects are utilized to store all candidate actions per agent. The list is created anew with each activation of the utility agent.

FiringTime – This class is used to create FiringTime objects. These objects are utilized to store the firing times that each action was "fired" (chosen and put into effect) per agent. As a new action fires, the list is updated.

PointUtility – This class is used to create PointUtility objects. These objects are utilized to store the point utility values that are awarded to each agent at various times during the simulation run. These values are discounted appropriately before storage.

SimpleUtilityAgentUmpire – This is the central class of the utility agent. All calculations and choice actions happen here. In this class, the following events take place:

The collection of data

The evaluation of data

The scheduling of the chosen action

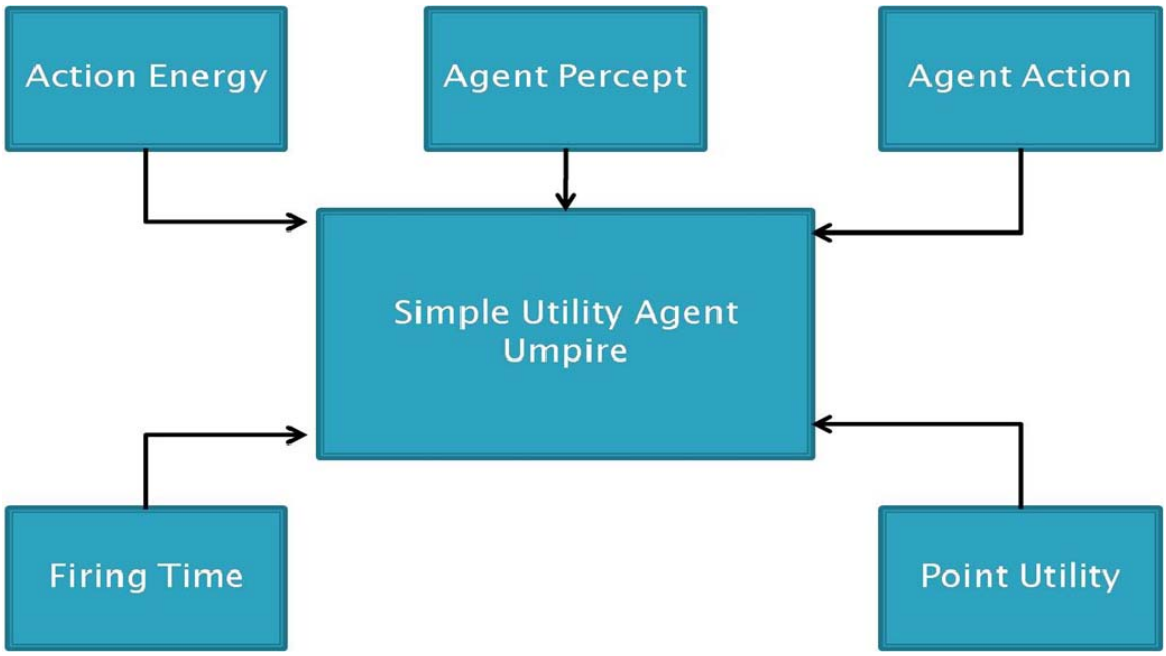


Figure 2. The classes of the utility-based agent and their interconnections

1. Collection of Data

During this procedure, we collect the population's average stance on security by going through each agent in the model that represents a population segment and adding his stance on security to a grand total, which we average in the end. Having already stored the average value from our previous collection event, we can easily calculate the change in the average stance on security by subtracting the current average from the previous average. The result of this calculation will be used as our "*point utility*" for this particular time in the simulation. A positive value of the point utility means that the average stance on security was decreased. This is a good result for our agent.

The next step is to discount this utility value in the appropriate way. This is a result of a procedure during which the utility value is discounted according to when this particular action was used in the past for the particular agent. The final discounted value of utility, called *Expected Utility*, is stored in a list of

ActionEnergy objects. This list contains the expected utility per action and per agent. This list is the final output of this procedure.

2. Evaluation of Data

During this procedure, the agent chooses between his six candidate actions according to the probability distribution specified by the Boltzmann distribution. The procedure is presented in more detail in Chapter II. The action that is selected, as a result of this procedure, is scheduled for firing with no time delay. Finally, the firing times list is updated with the new action that is being fired, and the point utility list for the particular agent is also updated by adding the action that was chosen along with the raw point utility value that was calculated during the collection of data. This procedure is repeated for each utility-based agent in the simulation.

3. Scheduling of Chosen Action

During this procedure, the chosen action is scheduled for execution in the simulation.

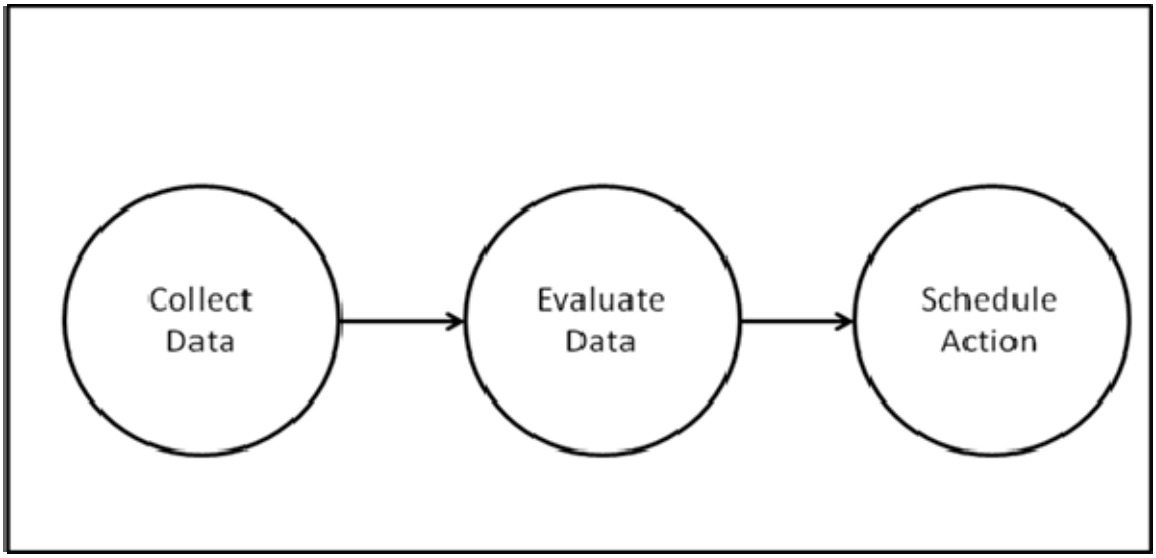


Figure 3. The sequence of events inside the main class of the utility-based agent

C. ADDITIONAL NEW COMPONENTS

The code for the utility-based agent became part of the CG model in version 0.7.1. The Excel configuration file was modified to reflect the changes in the code and to facilitate the scenario builder to create any scenario. A new tab was added to the Excel configuration file. Inside this tab, the user can enter all the information that the utility-based agent needs in order to function properly.

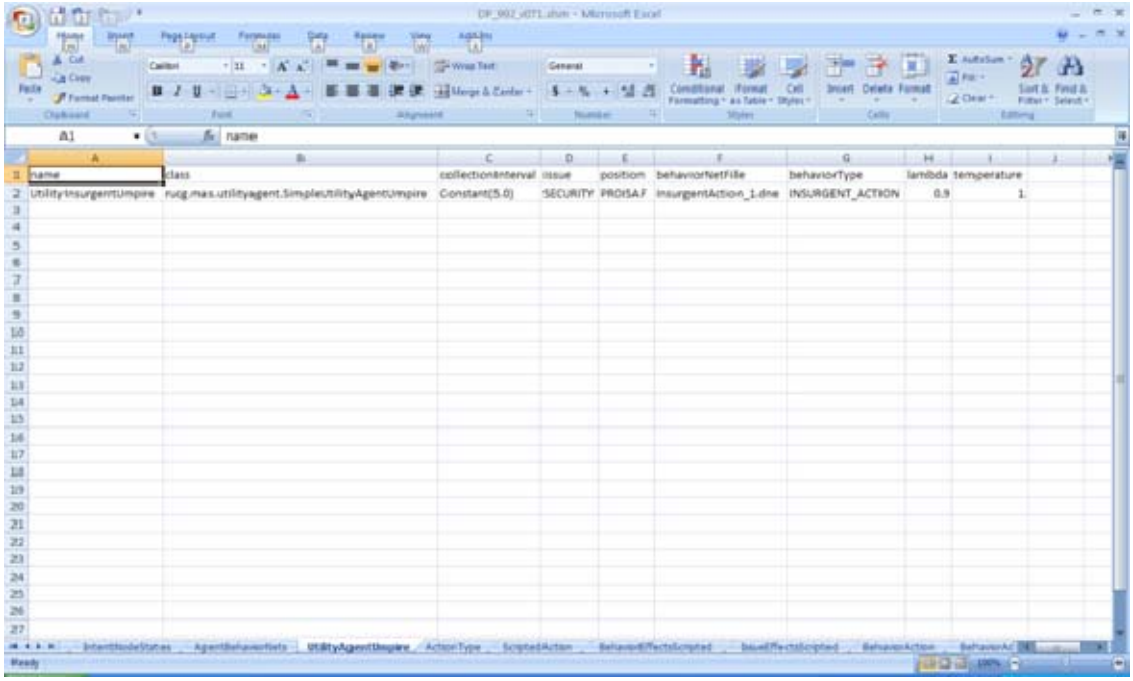


Figure 4. Snapshot of the new tab that supports the utility-based agent

Another new component is a logger called Action Activation Data Logger. This logger records the activation levels for all candidate actions each time the utility-based agent repeats its choosing procedure.

D. THE TEST RUN

To test the functionality of our new agent, we created a “blank scenario.” In this “blank scenario,” no other actor of the CG model is performing any actions, except the new utility-based agent. The possible action choices for our agent, independent from the state of the environment, are as follows:

- **AntiCoalitionForceMessage**: The insurgents spread to the population a rumor against the coalition forces.
- **AttackCoalition Force**: The insurgents conduct an attack on the coalition forces.
- **CivilianCasualty**: The insurgents attack a key civilian.
- **DamageInfrastructure**: The insurgents attack and cause damage to an infrastructure facility.
- **DoNothing**: Self explanatory.

We also prevented any consumption of consumables from all actors. In this way we eliminated any effects that the consumption might have on the population. Essentially, all infrastructure remained in a neutral state all throughout our "blank scenario." We performed two test runs, using two extreme values of temperature (0.1 and 1.0). During the execution of each test run, we recorded the activation levels for each candidate action. After gathering all data, we constructed two overlay plots, one for each value of temperature.

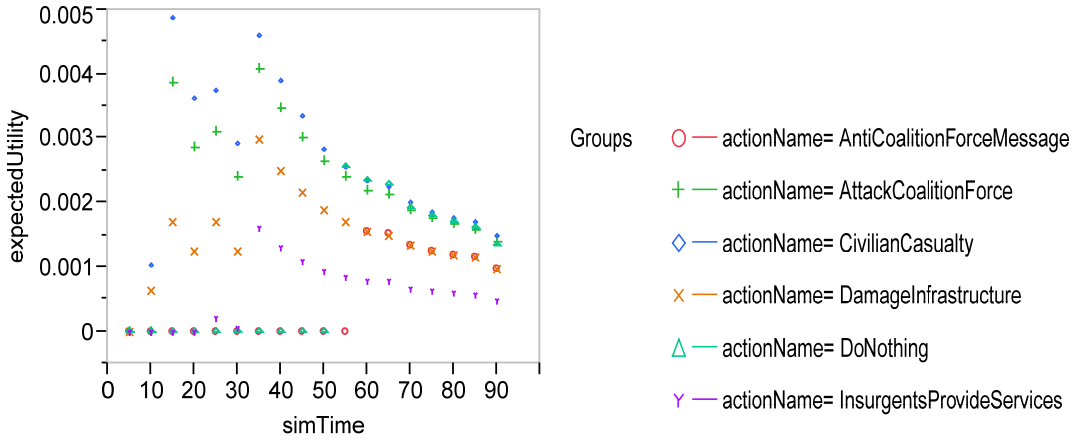


Figure 5. Overlay plot for temperature = 0.1

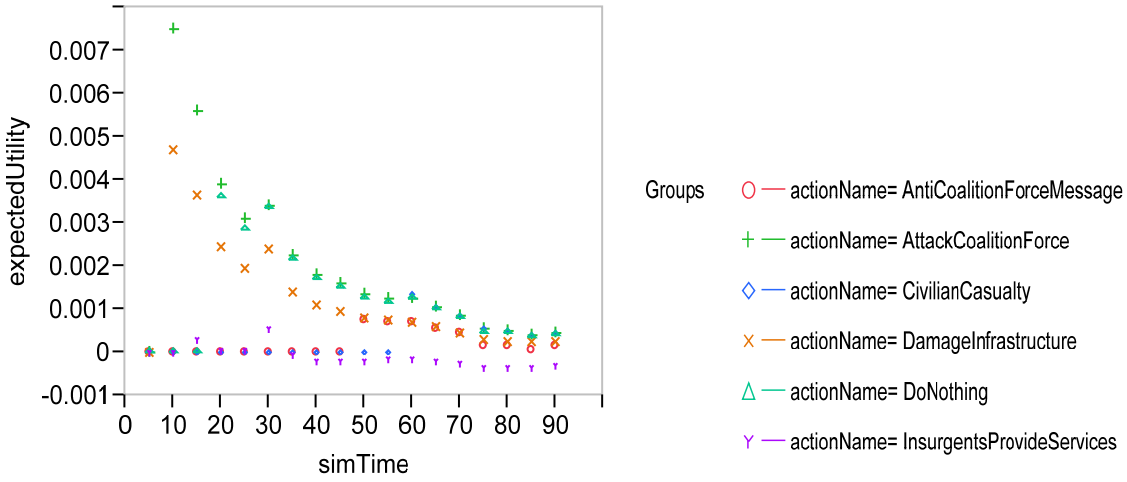


Figure 6. Overlay plot for temperature = 1.0

From these plots, we can see a difference in the way our agent chooses his actions between the two cases.

In the case with the lower temperature (greedy case), the agent is supposed to choose the action with the highest activation level. We can clearly see (Figure 5) that the utility value for most actions stays at a relatively high level until a simulation time of about 35. The agent does not care which action he chooses, as long as this action produces the best expected result for him. Eventually, all actions' activation levels grow smaller with time and they converge at a value right above zero. This can be explained by the fact that in our "blank" scenario no other actor does anything except our agents. Therefore, there are no other actions taking place that could influence the percepts our agent uses to calculate his utility value. With that in mind, it makes sense that the agent will collect the greatest amount of utility in the beginning of the simulation and, since nothing changes in his environment as the simulation progresses, his potential for any further improvements in his utility will gradually decrease until it reaches a value near zero. And this will eventually happen no matter which sequence of actions he might choose.

In the case with the higher temperature (exploration case), the agent might not necessarily choose the action with the highest activation level. This could lead, eventually, to a quicker conversion of the activation levels of all actions at a value close to zero (Figure 6). This can be explained by the fact that, in the exploration case, the agent does not favor one action over all others, thus leading to a more balanced choice between actions.

Since there are no other actions being performed by the other actors of the CG model in this "blank scenario," we observe the odd phenomenon of the steady decrease of the activation level for all actions. This absence of actions by all other actors in the model creates a "sterile" environment for our agent, thus not allowing us to draw any useful conclusions about the factors that contribute to the way our agent chooses his actions. What is necessary, in order to gain more insight on the way our agent works, is to put him in a fully active environment with other actors performing their own actions. We will examine such a scenario in Chapter IV. This test run was only supposed to test the functionality of our new agent and to give us a taste of the effects of temperature. In that regard, our test run was successful.

E. THE EVOLUTION

After the inclusion of the utility-based agent in the CG model, we tried to expand its applicability. We soon realized that, even though the creation of the agent was based on the insurgents component of the CG model, we could easily expand its use to the other components as well. All the scenario builder had to do was to identify the issue and position of interest, according to the agent he was describing, and enter the appropriate values in the Excel configuration file, as shown above (Figure 4).

Although the use of one percept per agent for the calculation of the utility value was appropriate, as a first step, the reality is that an agent might need more than one percept from its environment in order to formulate its utility value

and, eventually, determine its next actions. For that purpose, a new class was created. This new class allows the agent to use multiple percepts (if needed) for the calculation of its utility value:

AgentPercept – This class is used to create AgentPercept objects. These objects are utilized to store a list of percepts for each agent prototype.

The final utility value is the weighted average of all the percepts of the agent. The weight of each percept is provided by the user, thus making the utility-based agent essentially user driven.

It must be noted that, at this point, the use of multiple percepts is allowed only per agent prototype. This means that, if we allow the use of three percepts for the calculation of the utility of a certain agent prototype, all agents based on this prototype will use these three percepts in their calculations. There is no way that an agent of this particular prototype will use any other percepts.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. PROVING A POINT

A. INTRODUCTION

This chapter presents the effort that was conducted toward the verification of the new Cultural Geography (CG) model's functionality. We will begin with a brief description of the CG model and its components. Then we will provide a detailed account of the experiments that were conducted for the verification of the CG model's functionality. Finally, we will present the results of our analysis with the intention to prove that the agents inside the model behave in an acceptable way, thus answering the first of the two research questions posed in Chapter I ("Is reinforcement learning appropriate for use in social simulations and the CG model in particular?"). The second research question ("What are the advantages from the use of utility-based agents within the CG model?") will be answered in Chapter V, during the discussion of the analysis results.

B. THE CULTURAL GEOGRAPHY MODEL

TRAC Monterey developed the CG model in Java, using Simkit as the simulation engine. Alt, Jackson, Hudak and Lieberman (2009) described the CG model as a "re-usable framework for representation of the civilian population within an IW context" (p. 2). The reusability of the model is achieved through a modular approach in its design. The model attempts to represent Kilkullens's concept of "conflict ecosystem" that exists in an IW setting by utilizing a multi-agent system (MAS), as defined by J. Ferber (as cited in Valdez, 2009, p.9).

The basic components of the CG model are the following:

- Population
- Infrastructure
- Other actors

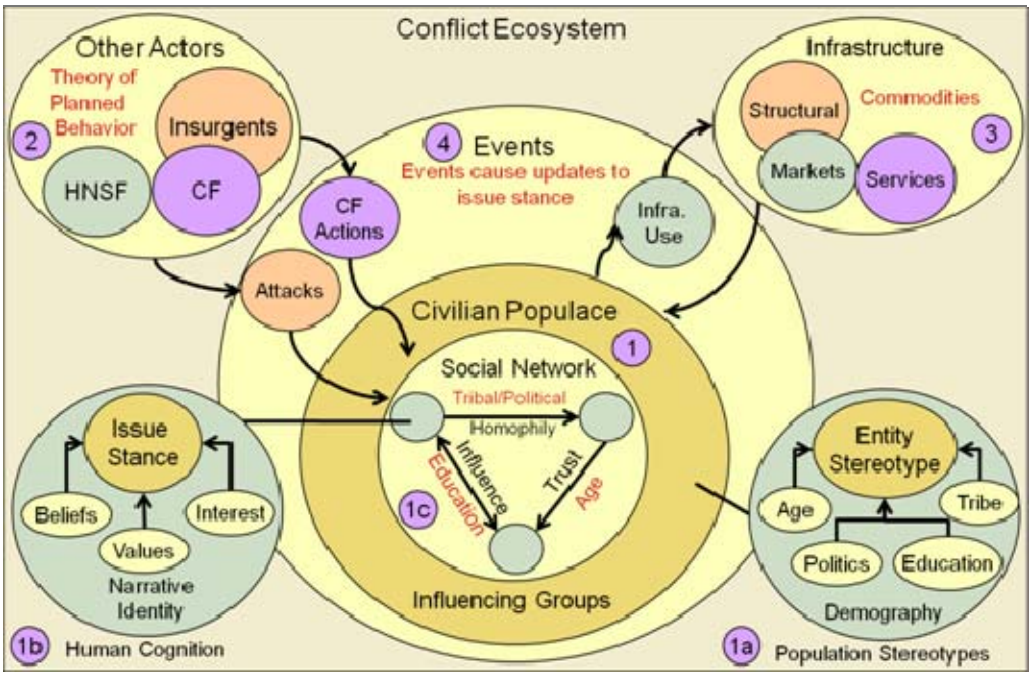


Figure 7. The Cultural Geography model (From TRAC Monterey)

All agents within the model are part of either the population or the "Other Actors" components shown in Figure 7. The infrastructure component is not represented by agents.

The primary forces that govern an agent's behavior within the CG model are the narrative paradigm and the theory of planned behavior (Alt et al., 2009). A brief description of these two concepts is provided, so the reader can form a general image of how the model works. We also talk briefly about the infrastrusture component of the model, describing the basic ideas behind this component.

1. Narrative Paradigm

The narrative paradigm is a theory developed by Fisher (1987). According to that theory, a narrative paradigm is "the incorporation of an entity's beliefs, values, and interests into a story, through which an agent evaluates the other stories of the world" (Valdez, 2009, p. 12). Alt et al. (2009) describe the

procedure that is utilized to construct an agent's narrative identity. The population in the area of interest is partitioned into entity types, according to relevant socio-demographic lines. These socio-demographic lines can be identified through the opinions of subject matter experts or polling data. For each entity type, a narrative identity is developed. Through a Bayesian network, these collections of beliefs and values are linked to the agent's stance on the issues of security, elections and infrastructure, thus constructing a Bayesian belief network (TRAC Monterey, 2009, p. 81). Each agent prototype has its own belief network that sets it apart from other agent prototypes. The Bayesian network that is used for infrastructure identification is shown in Figure 8:

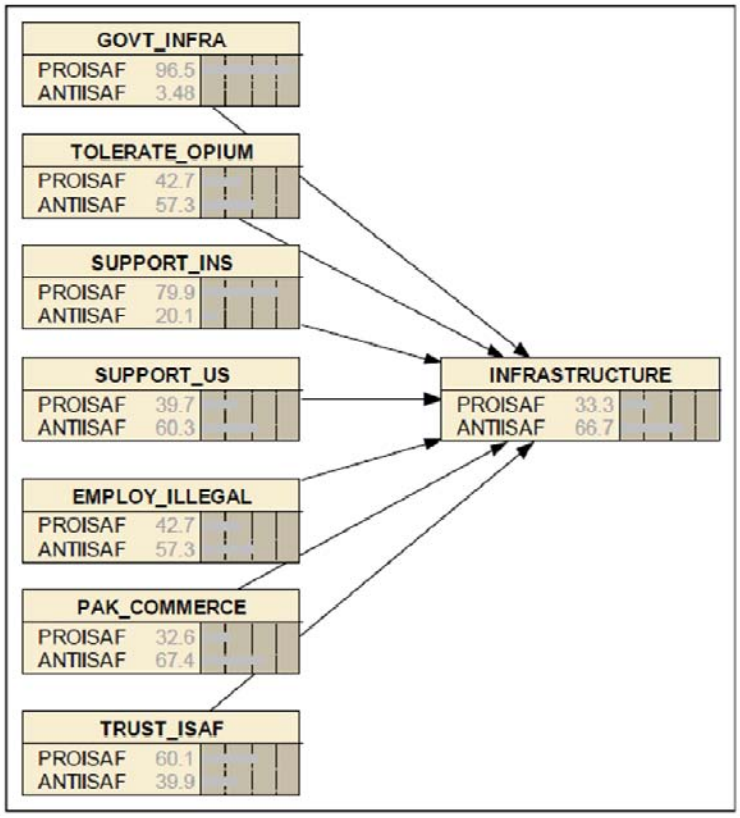


Figure 8. Bayesian network for Infrastructure (TRAC Monterey, 2009)

In this example, the indication "PROISAF" and "ANTIISAF" should be considered as a "YES" or "NO," respectively. For example, a value of "PROISAF"

in the TOLERATE_OPIUM behavior means that the agent tolerates opium trafficking. A value of “ANTIISAF” in SUPPORT_US means that the agent does not support the U.S.

Whenever an event happens that affects an agent prototype’s belief network, the values inside the network are updated accordingly. This results in an updated stance on security, elections and/or infrastructure. In addition, a “Homophily network” that represents “the likelihood of communication between two individuals in terms of their similarity among social factors” (Alt et al., 2009, p. 10) can also influence an agent’s beliefs and, eventually, his stance on the above-mentioned issues.

2. Theory of Planned Behavior

The theory of planned behavior is the main force behind an agent’s actions. As described by Alt et al., “individuals within a group will form an intention to adopt a behavior based on: 1) their attitude toward the behavior, 2) their perception of the group norms associated with that behavior, and 3) the individual’s perceived level of behavioral control in regard to that behavior” (2009, p. 5). The theory was implemented inside the CG model by utilizing a Bayesian network. An example of such a network is shown in the following figure. The work detailed in this thesis is designed to address deficiencies in the current implementation of the theory of planned behavior using Bayesian networks.

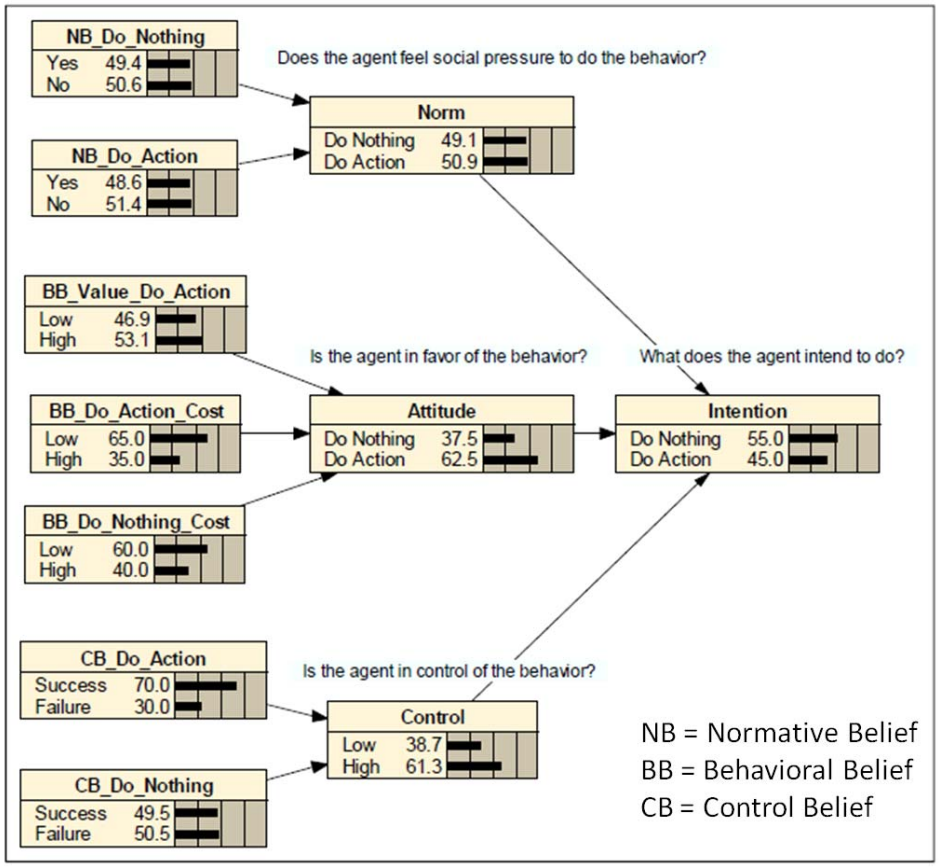


Figure 9. Theory of planned behavior network (From TRAC Monterey, 2009)

3. Infrastructure Component

Infrastructure is represented inside the model as multi-server queues. The agents can consume goods (food, water, etc.) or receive services (electricity, irrigation etc.). The agent's interaction with the infrastructure objects, combined with his narrative identity, can result in a change in the agent's stance on the issue of infrastructure. As is usually the case with multiple-server queues, we can establish limits to the server's capabilities. For example, the availability of a service can be increased, due to actions of friendly forces (construction), or decreased, due to the actions of enemy forces (attacks). In addition, a cost can be associated with the server's usage. For example, if a resource is located in

region A, an agent from region B can use this resource but he will have to pay a cost, whereas an agent located in region A will not have to pay this cost.

C. THE EXPERIMENTS

What follows is a description of the experimentation that took place with the CG model in order to monitor the agents' behavior and draw results. Each experiment is presented in the following way: a brief scenario description, the independent variables (the factors that were manipulated), the dependent variables (the measures of effectiveness that were measured), constraints, limitations and assumptions (as necessary), and a description of the results along with tables and diagrams as deemed appropriate. All of the analysis was done using the JMP analysis software. The scenarios were developed in collaboration with TRAC Monterey.

1. About the Scenarios

All the scenarios of this study deal with a province of Afghanistan called Kandahar. More specifically, the population of this region was segmented into factions according to the following characteristics:

- Family/Clan status
- Tribe
- Disposition
- Political affiliation
- Age – Gender

The specific values that were used for these characteristics are shown in the following table:

Table 1. Demographic characteristics used for the segmentation of the population

Family/Clan Status	Tribe	Disposition	Political Affiliation	Age / Gender
Inherited	Empowered (Barakzai, Popalzai, Mohammadzai)	Urban	Anti-Government	Military Age Male
Achieved	Passive (Alokozai, Noorzai)	Kuchi	Neutral	Elder Male
Poor / Unemployed	Marginalized (Noorzai, Ishaqzai, Alizai, Ghilzai)		Pro-Government	Military Age Female
				Elder Female

Through consultation with experts in the region, the scenario building team examined all possible combinations of the demographic characteristics and narrowed the resulting population groups to 15. For each one of those groups, a narrative identity was created and was inserted in the model.

All agents representing the population are also infrastructure consumers, following the general rules described in paragraph B3, above.

For the purposes of our study, we will examine the population in an aggregate way, so there is no need to analyze the segmentation process in much more detail.

2. Variables

Independent Variables

In all of the scenarios, we manipulated one or more of the following three factors:

- **Lambda**: As previously described in Chapter II, this variable is the discount rate for percepts that will be received in the future. A discount rate closer to 0 drives the agent to maximize his short term reward by not giving too much value to future rewards. From this point on, we will refer to a rate closer to 0 as “short term memory.” In contrast, a discount rate closer to 1 drives the agent to maximize his long term reward. We will refer to such a rate as “long term memory.”

- **Temperature**: As described in Chapter II, the value of temperature defines the "greediness" of the agent. A value closer to 0 drives the agent to make greedy decisions (exploit), whereas a high temperature value allows more diversity in the agent's decision making (explore).
- **Collection interval**: This variable shows the amount of simulation time steps that elapse between two decisions of an agent.

Dependent Variables

For all scenarios, we measured the population's stance on the issues of security, infrastructure and governance. Since we chose to examine the functionality of an insurgent agent, we used as our main measure of effectiveness the population's stance on security. We made this decision because, according to all scenarios, the stance on security is the percept that the insurgent agent uses in order to calculate his utility values and, eventually, formulate his decisions.

3. General Constraints, Limitations and Assumptions

a. Constraints

There is limited time in which to conduct this study, constrained by school requirements.

b. Limitations

- The population is represented by a total of 350 agents.
- The other actors (Insurgents, Government of the Islamic Republic of Afghanistan (GIRoA), Afghanistan National Security Forces (ANSF), International Security Afghanistan Forces (ISAF)) are represented by one agent per actor and per region, a total of 20 agents.
- A small sample of experts provided survey input regarding the impact of events/themes on population beliefs.

c. Assumptions

- 350 agents representing the major population groups within Kandahar Province provide sufficient fidelity to extract population's beliefs and stances on issues.
- Fully vetted expert input, with multiple years of experience concerning Kandahar, adequately represents the impact of events on the population identity groups.

4. Candidate Actions

The focus of our study will be on the agents that represent the insurgents inside the model. There is one agent per region, bringing the total of the insurgent agents to five. Each agent must choose among the following actions:

- **DoNothing**: Self-explanatory. The agent performs no action.
- **KillCivilServant**: The agent makes an assassination attempt against a Civil Servant.
- **IED**: The agent plants an IED against any target.
- **IED_ANSF**: The agent plants an IED targeting the ANSF forces.

5. The Experimental Design

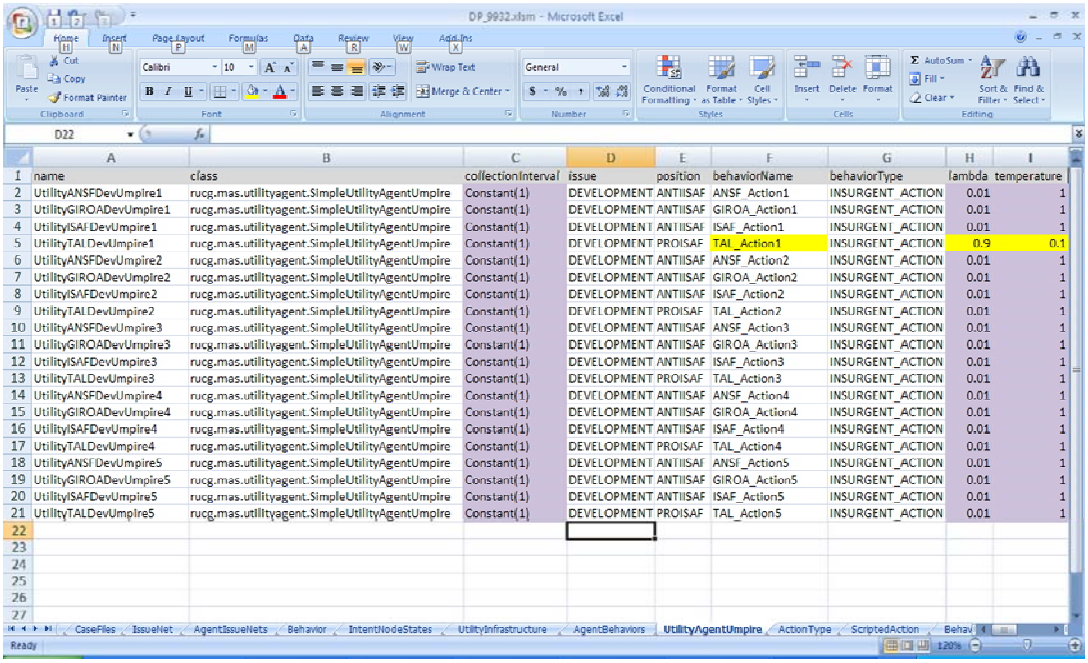
We conducted our research either by performing single runs of our scenarios or by doing a design of experiments based on these scenarios. In the cases in which we used a design of experiments, the method used was the Near Orthogonal Latin Hypercube (NOLH). This design allows us to fully explore the factor space and, thus, achieve approximate orthogonality of input factors. By using the NOLH method, the experimental design points form a representative subset of the hypercube of explanatory variables (Alt et al., 2009). The development of the design points was done by utilizing the Design of Experiments tool that was developed by TRAC Monterey. This tool creates design points based on the NOLH method and incorporates them into the scenario files that are used by the CG model.

D. THE RESULTS

In the following paragraphs, we will give a detailed account of the results of our statistical analysis. The results will be presented in such a way that they answer the research questions posed in Chapter I. All the general information detailed in section C above pertains to all scenarios. Any additional scenario-specific information is mentioned in each scenario paragraph as needed.

1. Scenario 1 – Simple Run

For this scenario, we controlled the collection interval variable by setting it to a value of 1 for every agent. Moreover, we gave to every agent (except Tal1) a value of 0.01 for lambda and a value of 1 for temperature. For the agent Tal1 (who represents an insurgent agent in the region of Kandahar City (KC)) we gave a value of 0.9 for lambda and a value of 0.1 for temperature. These settings were inserted into the Excel scenario setup file as shown in Figure 10:

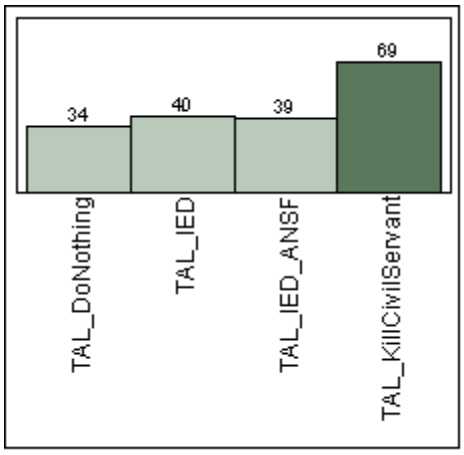


	name	class	collectionInterval	issue	position	behaviorName	behaviorType	lambda	temperature
2	UtilityANSFDevUmpire1	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ANSF_Action1	INSURGENT_ACTION	0.01	1
3	UtilityGIROADevUmpire1	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	GIROA_Action1	INSURGENT_ACTION	0.01	1
4	UtilityISAFDevUmpire1	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ISAF_Action1	INSURGENT_ACTION	0.01	1
5	UtilityTALDevUmpire1	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	PROISAF	TAL_Action1	INSURGENT_ACTION	0.9	0.1
6	UtilityANSFDevUmpire2	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ANSF_Action2	INSURGENT_ACTION	0.01	1
7	UtilityGIROADevUmpire2	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	GIROA_Action2	INSURGENT_ACTION	0.01	1
8	UtilityISAFDevUmpire2	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ISAF_Action2	INSURGENT_ACTION	0.01	1
9	UtilityTALDevUmpire2	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	PROISAF	TAL_Action2	INSURGENT_ACTION	0.01	1
10	UtilityANSFDevUmpire3	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ANSF_Action3	INSURGENT_ACTION	0.01	1
11	UtilityGIROADevUmpire3	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	GIROA_Action3	INSURGENT_ACTION	0.01	1
12	UtilityISAFDevUmpire3	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ISAF_Action3	INSURGENT_ACTION	0.01	1
13	UtilityTALDevUmpire3	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	PROISAF	TAL_Action3	INSURGENT_ACTION	0.01	1
14	UtilityANSFDevUmpire4	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ANSF_Action4	INSURGENT_ACTION	0.01	1
15	UtilityGIROADevUmpire4	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	GIROA_Action4	INSURGENT_ACTION	0.01	1
16	UtilityISAFDevUmpire4	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ISAF_Action4	INSURGENT_ACTION	0.01	1
17	UtilityTALDevUmpire4	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	PROISAF	TAL_Action4	INSURGENT_ACTION	0.01	1
18	UtilityANSFDevUmpire5	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ANSF_Action5	INSURGENT_ACTION	0.01	1
19	UtilityGIROADevUmpire5	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	GIROA_Action5	INSURGENT_ACTION	0.01	1
20	UtilityISAFDevUmpire5	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	ANTIISAF	ISAF_Action5	INSURGENT_ACTION	0.01	1
21	UtilityTALDevUmpire5	rucg.mas.utilityagent.SimpleUtilityAgentUmpire	Constant(1)	DEVELOPMENT	PROISAF	TAL_Action5	INSURGENT_ACTION	0.01	1

Figure 10. Setup of variables for all acting agents

With these values, we expect the agent Tal1 to act in a greedy way and have a long-term memory. All other agents are expected to act in a more exploratory way and have a short-term memory. In addition to that, we gave to the action KillCivilServant an enhanced reward value of 100 and kept the reward value of all other candidate actions at a value of 1. This enhancement was done only for the agent Tal1. By doing that, we expected a greedy agent (like Tal1) to choose the action with the enhanced reward more often than all other candidate actions. The scenario ran in one replication.

The distribution of the chosen actions for Tal1 was:



Frequencies

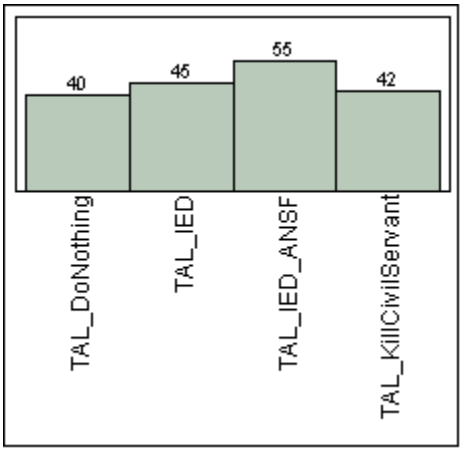
Level	Count	Prob
TAL_DoNothing	34	0.18681
TAL_IED	40	0.21978
TAL_IED_ANSF	39	0.21429
TAL_KillCivilServant	69	0.37912
Total	182	1.00000

N Missing 0

4 Levels

Figure 11. Distribution of actions for agent Tal1 (scenario 1)

The distribution of actions for agent Tal2 was:



Frequencies

Level	Count	Prob
TAL_DoNothing	40	0.21978
TAL_IED	45	0.24725
TAL_IED_ANSF	55	0.30220
TAL_KillCivilServant	42	0.23077
Total	182	1.00000

N Missing 0

4 Levels

Figure 12. Distribution of actions for agent Tal2 (scenario 1)

By examining these two figures, it is clear that the agent Tal1 favors the KillCivilServant action much more than Tal2 does. However, is this difference in action choices a significant one? By performing a contingency analysis between the action choices of the two agents, we tried to determine whether the difference in the action choices between the two agents was significant or not. With an alpha level of 0.05, here are the results of this analysis:

Table 2. Contingency table of the candidate actions by agent

		state				
	Count Total % Col % Row %	TAL_DoNothing	TAL_IED	TAL_IED_ANSF	TAL_KillCivilServant	
entityName	TAL_1	34 9.34 45.95 18.68	40 10.99 47.06 21.98	39 10.71 41.49 21.43	69 18.96 62.16 37.91	182 50.00
	TAL_2	40 10.99 54.05 21.98	45 12.36 52.94 24.73	55 15.11 58.51 30.22	42 11.54 37.84 23.08	182 50.00
		74 20.33	85 23.35	94 25.82	111 30.49	364

Tests

N	DF	-LogLike	RSquare (U)
364	3	5.0759665	0.0101

Test	ChiSquare	Prob>ChiSq
Likelihood Ratio	10.152	0.0173*
Pearson	10.072	0.0180*

Figure 13. ChiSquare test for the likelihood of action choice occurrence

Both Figure 13 and Table 2 showed that the difference between the action choices of Tal1 and Tal2 is significant, with a probability that the difference in action choice happens by pure chance at a value of 0.0173.

The next thing we wanted to examine was how the agent Tal1 made his choices over time. We already knew that he favored the KillCivilServant action over all others. We wanted to see how these choices were distributed over time and how these compare to the distribution of choices of Tal2 for this action. To accomplish that, we calculated the moving average of the number of times the action KillCivilServant for agents Tal1 and Tal2 were chosen and we constructed an overlay plot to better illustrate any differences between agents Tal1 and Tal2. The plot looks like this:

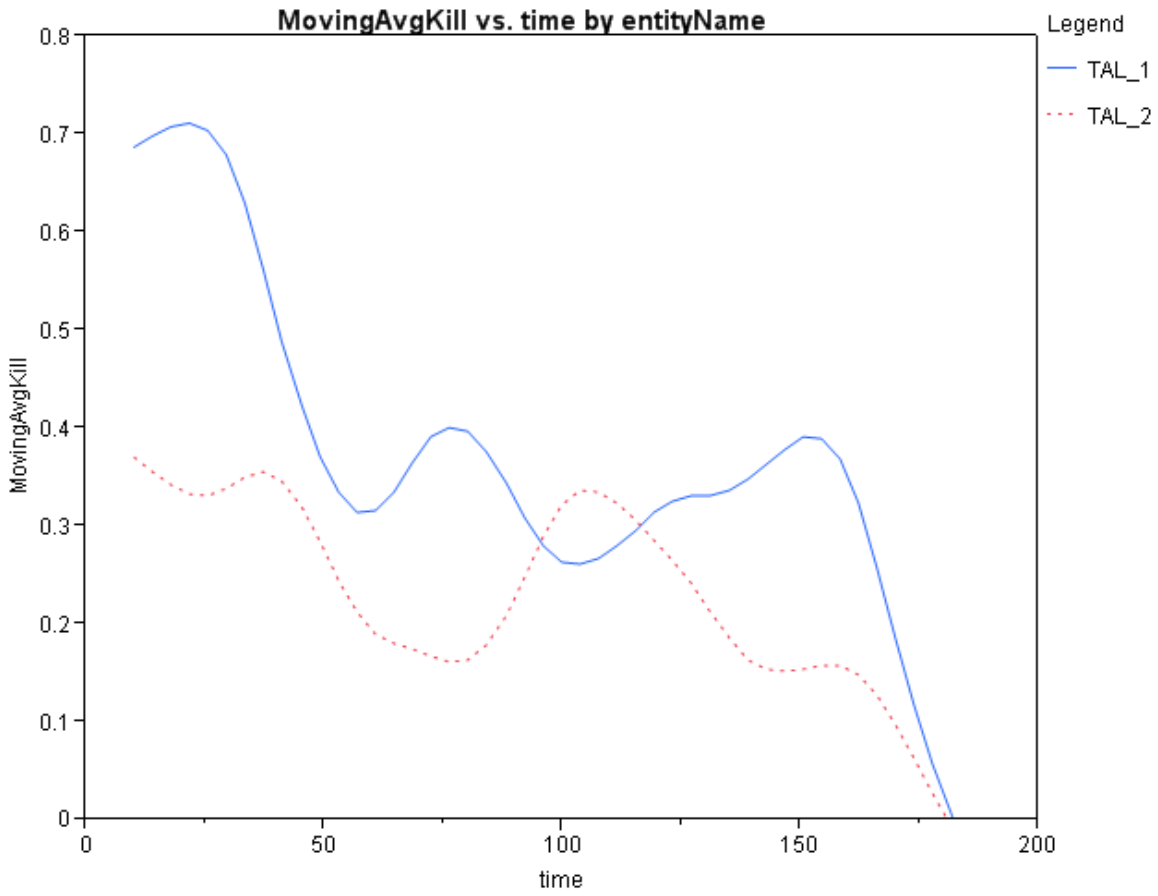


Figure 14. Moving averages of agents Tal1, Tal2 over time for action KillCivilServant (Scenario 1)

For the plot shown above, a value of 0.7 for the moving average at time 25 means that in the previous ten time steps (between time steps 16 and 25) the action KillCivilServant was chosen seven times out of ten. Note that the action is selected much less frequently over the course of the run. Since the utility is based on the change in population stance from one observation to the next, as the changes in population stance go down, the activation levels do as well. In the current model, the population's stance tends to converge over time and each subsequent action has less ability to move the issue stance. This results in a reduced effectiveness for all action choices by the end of the scenario run. This is a known issue with the model's representation of issue stance, and future work is planned to address this.

So far, we have shown that the agent Tal1 works properly inside the CG model-making choices. More specifically, by enhancing the reward value of one candidate action over the others for one particular agent, we showed that this agent favors this action in a significant way, compared to other insurgent agents operating in neighboring regions. This is a strong indication that our agent makes effective use of reinforcement learning during the procedure of choosing his next action. Finally, we showed that this favoring is spread throughout the duration of the simulation and is not happening only at the beginning or at the end of the simulation run. Note that the impact of any action choice by actors late in the run results in a lower impact on the population, and lower utility, due to the convergence of the population's issue stance over time.

2. Scenario 1 – Experimental Runs

For this run, we used the settings of the previous scenario. We designed an experiment by using the NOLH design for optimality. By varying the lambda and the temperature variables between the values of 0 and 1, we ended up with 33 design points. The scenario ran for 10 replications. The total number of runs for our experiment was 330. The runs were performed in TRAC Monterey, using the computers located in the Conference Room lab.

The design points and the respective values for lambda and temperature are shown in the following table:

Table 3. Design points for experimental design

DP	Lambda	Temperature
1	1	0.18
2	0.92	1
3	0.89	0.49
4	0.61	0.89
5	0.94	0.13
6	0.97	0.94
7	0.72	0.52
8	0.58	0.72
9	0.69	0.33
10	0.78	0.69
11	0.75	0.3
12	0.8	0.75
13	0.63	0.24
14	0.86	0.63
15	0.66	0.27
16	0.83	0.66
17	0.55	0.55
18	0.1	0.92
19	0.18	0.1
20	0.21	0.61
21	0.49	0.21
22	0.16	0.97
23	0.13	0.16
24	0.38	0.58
25	0.52	0.38
26	0.41	0.78
27	0.33	0.41
28	0.35	0.8
29	0.3	0.35
30	0.47	0.86
31	0.24	0.47
32	0.44	0.83
33	0.27	0.44

At first, we examined our MOE (population stance on security for the Kandahar City location). We averaged our MOE across replications and design points. This is the plot of our averaged MOE over simulation time:

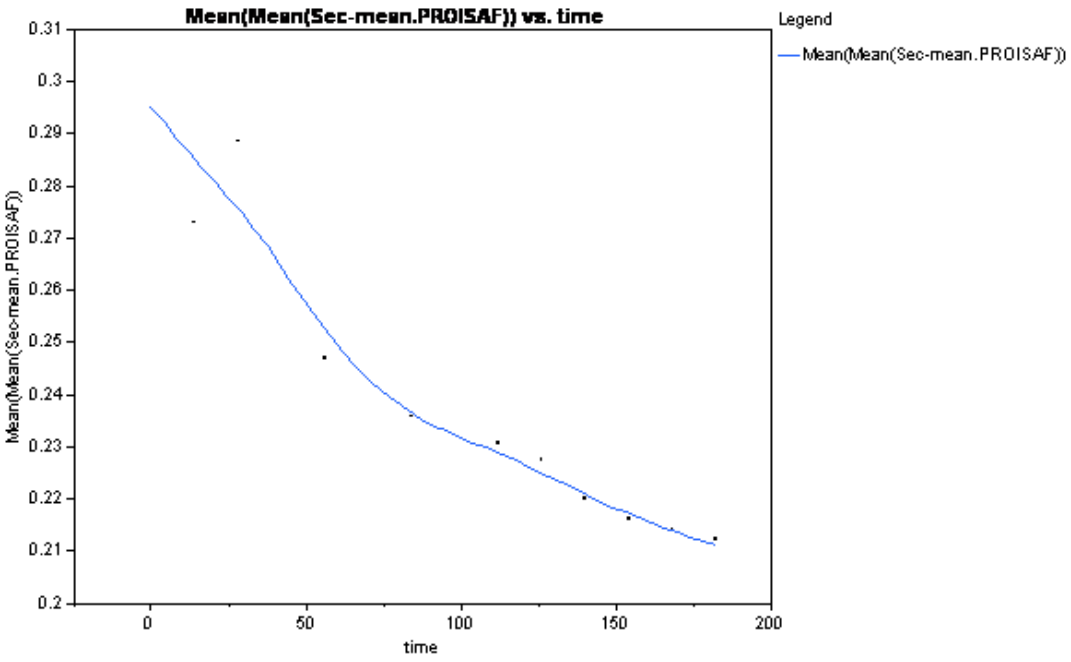


Figure 15. Plot of the mean population stance on security over simulation time

Since the purpose of our agent is to decrease the population's satisfaction with security in the area over time, it appears as if our agent was successful.

To get an idea of which combination of lambda and temperature produced the most optimal results, we constructed a contour plot of our MOE over lambda and temperature.

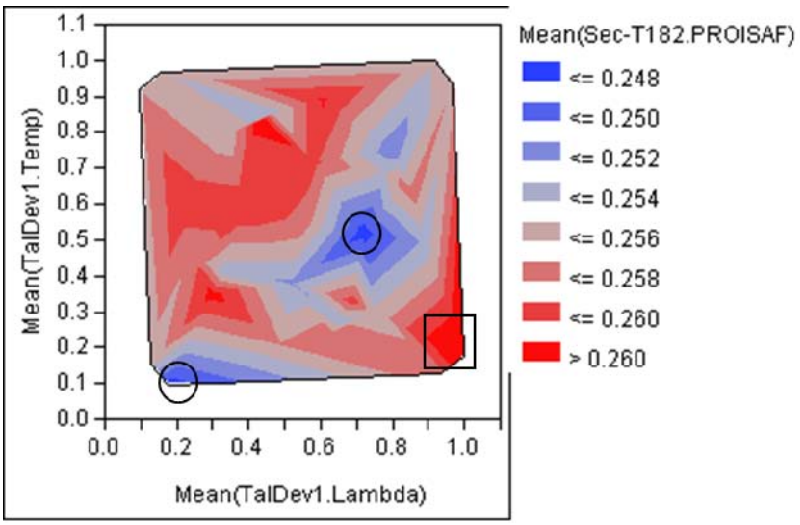


Figure 16. Contour plot of the population's stance on security over lambda and temperature

Since we were interested in the lowest values of our MOE, we focused our attention on the areas within the two circles. The combinations of lambda and temperature within these two circles produce the greatest decrease in the population's satisfaction, the goal of our agent. By examining the Table 3, we can determine that these design points are DP19, with lambda = 0.18 and temperature = 0.1, and DP7, with lambda = 0.72 and temperature = 0.52. We will compare these design points with DP5, which seems to produce the worst results, indicated in Figure 16 with a square.

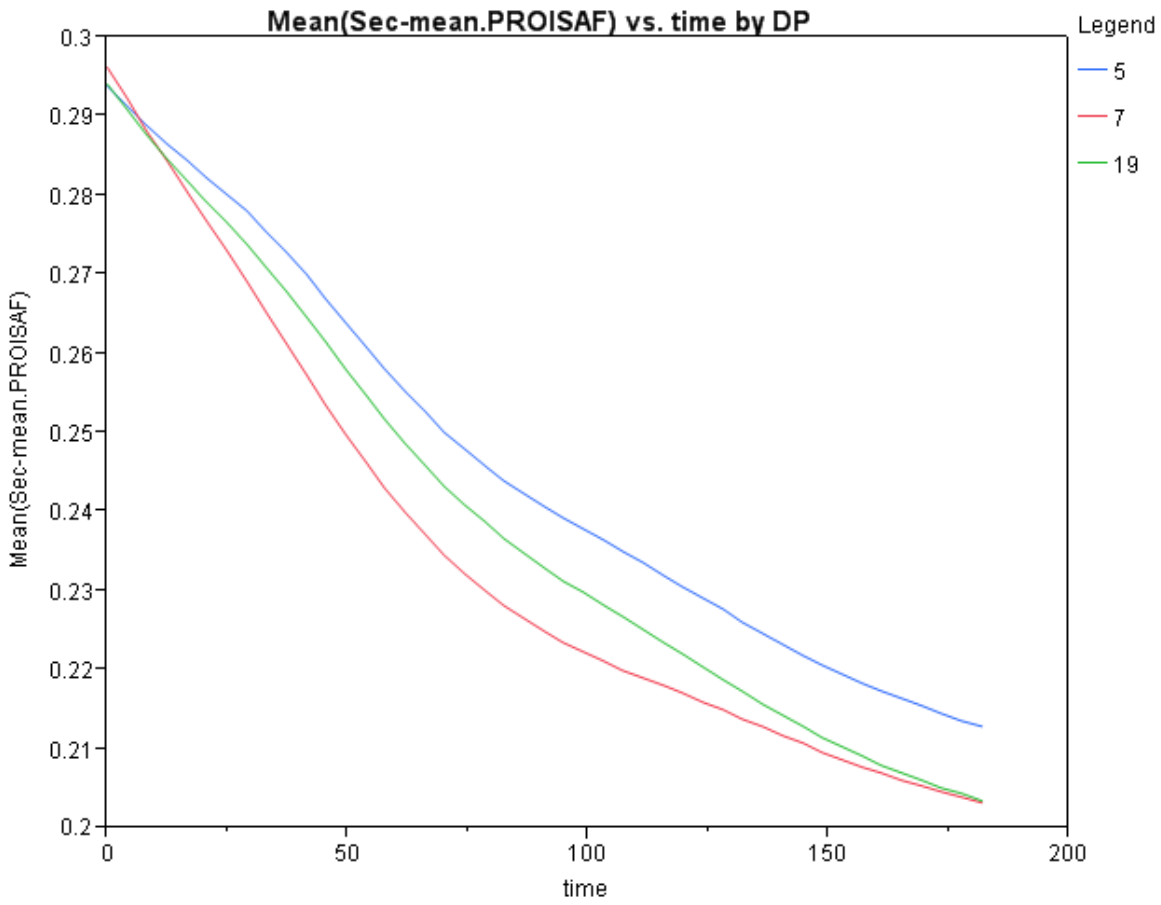


Figure 17. Overlay plot of Population's stance on Security over time by design point

From Figure 17, we can see that among the design points in our contour plot, DP7 and DP19 (represented by the bottom two lines in the plot) produce better results in less time, compared to design point DP5 (represented by the top line in the plot) which also produces good results but in a much slower way.

3. Scenario 2 – Experimental Runs

For the purposes of this experiment, we constructed a scenario that treated all candidate actions for all acting agents in a similar manner. This means that all candidate actions have the same weight. No action is favored over any other. In each region, we placed one agent per category (GIROA, ANSF, ISAF, TAL). We focused our study in the region of Kandahar City. The factors we

examined for this scenario are the three factors that were mentioned in paragraph C2 of this chapter. These factors were the collection interval, the lambda and the temperature. We varied the values of lambda and temperature from 0 to 1. We varied the value of collection interval from 1 to 5. We did this for all acting agents in the model. By using the design of experiments tool built by TRAC Monterey, we constructed a design of experiments, based on the NOLH method, ending up with 512 design points. The simulation ran for five replications, bringing the total number of runs to 2,560.

After averaging across design points, we performed a standard least squares regression, using the population's stance on security as the dependent variable and our three factors per acting agent (a total of 12 factors) as the independent variables. In that way, we wanted to examine which of the above-mentioned factors contributes significantly to our MOE. Here are the results of our analysis at an alpha level of 0.05:

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	0.238073	0.002565	92.83	0.0000 *
CollIntGIROADev1	0.0009983	0.000278	3.59	0.0003 *
CollIntANSFDev1	-0.001794	0.000278	-6.46	<.0001 *
CollIntISAFDev1	-0.011247	0.000278	-40.49	0.0000 *
CollIntTALDev1	0.0074984	0.000278	27.00	<.0001 *
LambdaGIROADev1	-0.000744	0.001235	-0.60	0.5470
LambdaANSFDev1	0.0012615	0.001235	1.02	0.3072
LambdaISAFDev1	-0.001114	0.001235	-0.90	0.3670
LambdaTALDev1	-0.000328	0.001235	-0.27	0.7909
TempGIROADev1	0.0001838	0.001235	0.15	0.8817
TempANSFDev1	-0.001469	0.001236	-1.19	0.2344
TempISAFDev1	0.000762	0.001235	0.62	0.5374
TempTALDev1	-0.000405	0.001235	-0.33	0.7430

Figure 18. Analysis results for scenario 3 experimental run

Figure 18 shows clearly that the collection interval contributes more significantly than any other factor to our MOE. This happens for all acting agents, no matter the role they play in the simulation.

Focusing our attention on the collection interval factor, we can examine which values of it produce the best results. For that purpose, we constructed plots of our MOE over each of the acting agents' collection interval for the region we are examining (Kandahar City).

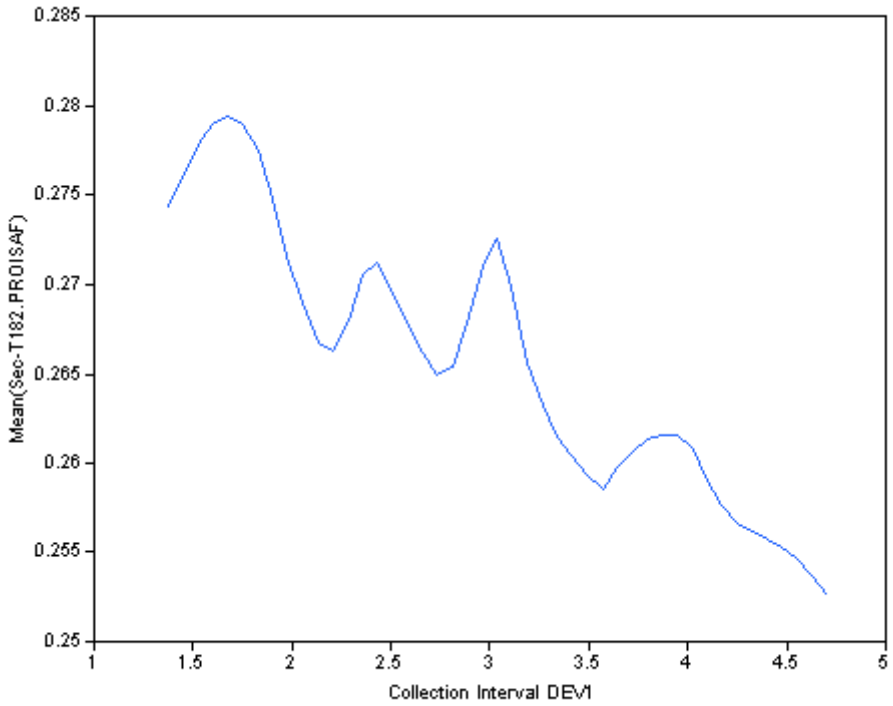


Figure 19. Plots of the population's stance on security over the acting agents' collection intervals

The plot shows that better results were produced when the collection interval had a high value. This means that when our agent allows more time between his action choices, the actions he chooses produce better results.

The analysis performed on this scenario showed that the collection interval variable outshines lambda and temperature in significance, when examined together. This is the reason why, in our previous scenarios, we controlled the collection interval variable in order to isolate the effects of the lambda and temperature variables and examine how these variables affect the action choices of our agents and, in extension, our MOE.

THIS PAGE INTENTIONALLY LEFT BLANK

V. FINAL THOUGHTS AND A LOOK TO THE FUTURE

A. INTRODUCTION

The concluding chapter of this study will begin with a brief discussion of the major findings of our analysis results, in which we will attempt to show the benefits of this study. Finally, we will present our suggestions for future study on the application of reinforcement learning within the Cultural Geography (CG) model.

B. DISCUSSION OF ANALYSIS RESULTS

To better illustrate the results of our analysis, it is deemed appropriate to associate them with the research questions we posed in Chapter I. The two questions were:

- Is reinforcement learning appropriate for use in social simulations and the CG model in particular?
- What are the advantages of the use of utility-based agents within the CG model?

To answer the first question, we tried to show:

- That the learning agent prototype we developed was functioning properly
- That it was producing the expected results within the simulation.

To show that the learning agent was functioning properly, all we had to examine was whether it was using reinforcement learning for its action decisions. We focused on one agent in a specific region and enhanced the utility value of one of its candidate actions over all others. Moreover, we made our agent greedy, so that it would favor that action over all its other candidate actions. We kept our third independent variable, the collection interval, constant for all agents in our scenario. The results showed that our agent preferred the action with the enhanced utility throughout the duration of the simulation run. When compared to

another agent in the same region, who was not greedy and did not have one of its candidate actions blessed with enhance utility, we saw that the difference in the two agents' action choices was a statistically significant one. This is a strong indication that our learning agent functions using the principles of reinforcement learning.

After performing a design of experiments on the same scenario, we saw that our learning agent produces the desired result, namely a decrease in the population's mean stance on security. Moreover, by constructing a contour plot of our MOE over lambda and temperature, we had the opportunity to isolate the combinations of lambda and temperature that contribute to the best and the worst results. A comparative analysis of these two combinations showed that they both produce the desired result, but the combination that produces the best results does so in a much faster way. A calculation of that difference showed that, by choosing the right combination of lambda and temperature, we gain the desired results about 12% faster, on average.

Finally, we performed a design of experiments based on a more general scenario, one that treated all agents and their candidate actions in the same manner. In that experiment, we examined the impact of all three independent variables (lambda, temperature and collection interval) on our chosen MOE. The results showed that the collection interval plays the most significant role in determining the value of our MOE, far outshining any effects the other two variables might have. By further examining the effects of the collection interval, we discovered that the best results are produced when the collection interval has a high value. This makes intuitive sense because when an agent allows for more time to elapse between his action choices, it can allow its previous action to better show its effects, and it can also gather more information from its environment in order to make its next decision as good as possible.

The answer to the second question is not the result of any statistical analysis, but stems from the experience we gained by working with our learning agents within the CG model. Our conclusions about this research question can be summarized in the following:

- The agents behave in a more realistic way. In the previous incarnation of the CG model, the agents performed their actions according to a predetermined plan that was hard coded inside the scenario file. This fact did not allow the agent to assess the situation and act accordingly. No matter what was happening in the world around it, the agent would perform its predetermined actions. With reinforcement learning, the agent gathers information from its environment and uses it to formulate an action plan. By constantly reexamining its past actions and their results, the agent evaluates all candidate actions and eventually decides on the course of action that best serves its interests.
- The implementation of reinforcement learning in the CG model allows for more flexibility for the user. Through a small number of parameters, the user can change the way the agents behave and, by doing so, explore a potentially infinite number of action sequences. Moreover, the setup time for the scenarios becomes considerably easier and faster, since the user does not have to create pre-scripted actions for each acting agent.
- Our experimentation showed that the scenario execution time of the CG model with reinforcement learning becomes considerably lower, when compared to the time it takes to run a scenario with hard coded actions, thus allowing the users to make better use of their computer resources by adding more agents in their scenarios and modeling situations that are more complex. At first, this might sound counter intuitive. How can a scenario that makes the agents, choose and then execute their actions, run faster than a scenario that only makes agents execute pre-scripted actions, and there is no choosing involved? To answer this question, we must take a look at what goes on inside the model during execution time. In the older version of the CG model, the agent had to access the model's Bayesian network through Netica. Netica is an application, separate from the CG model, but running in parallel, which handles all computations involved during the execution of each action as well as the updating of the population's stances, after the execution of each action. Netica was called even during scenarios with scripted actions to control action selection related to infrastructure objects. Each agent within the model possessed a separate behavior network for each commodity provided by the infrastructure servers

as well as any other actions they might have. In the new version of the CG model, the necessary computations during the execution of each action, as well as all action selections related to infrastructure, happen inside the model, thus resulting in considerable gains in execution time, since it is no longer necessary to call a separate application.

- In the new version of the CG model, all actions performed by the agents can be traced. This facilitates analysis by potential users to gain an understanding of the impact of agent actions, facilitating course of action analysis.

C. FUTURE WORK

There are many potential avenues a researcher could explore, using this study as a starting point. Some of these areas are illustrated below:

- **Use of another action choice algorithm**: Instead of using the softmax algorithm for action selection, it would be interesting to utilize another algorithm, for example the ε-greedy algorithm, for the action selection process. A comparative analysis with the softmax action selection algorithm could prove to be quite interesting.
- **Establishment of a schedule for the gradual decrease of temperature over time**: In the implementation of the softmax algorithm presented in this study, the temperature is being held constant throughout the scenario execution. One possible opportunity for research could be the creation of a function that would decrease the value of temperature as the scenario advances. In that way, we could make the agents move from exploration to exploitation in a controlled way, similar to that of the simulated annealing search algorithm (Russell & Norvig, 2003).
- **Analysis of the impact of reinforcement learning in other components of the CG model**: In this study, we focused our analysis on the impact that reinforcement learning has on the population's stance on security. There are many other components in the CG model that could be explored in order to determine the impact of reinforcement learning on them. For example, one study could examine the impact of reinforcement learning on infrastructure. More specifically, how the implementation of reinforcement learning in the infrastructure consumption chain affects the action choices of the consumers.
- **Allow the agent to use multiple percepts for his utility calculation**: The implementation presented in this study allows the agent to use only one percept for his utility calculation. Although

this is acceptable as a first step, it cannot be considered as a final solution. A more realistic approach would be to allow the agent to use multiple percepts for his utility calculation. These percepts would be user defined and would apply to all agents of the same agent prototype. It should be noted that the infrastructure for this implementation is already in place within the CG model. Only minor changes in the reinforcement learning algorithm code are required for making the proposed approach a reality.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Alt, J., Jackson, L., Hudak, D., & Lieberman, S. (2009). *The Cultural geography model: Evaluating the impact of tactical operational outcomes on a civilian population in an Irregular Warfare environment*. Unpublished manuscript.

Fisher, W. (1987). *Human communication as narration: Toward a philosophy of reason, value, and action*. Columbia, SC: University of South Carolina Press.

Perkins, T., Pearman, G., & Jackson, L. (2009). *Cultural geography data development.* Technical report submitted for publication, TRAC Monterey, Monterey, CA.

Russel, S., & Norvig, P. (2003). *Artificial Intelligence a modern approach* (2nd ed.). Upper Saddle River, NJ: Pearson Education.

Sutton, R., & Barto A. (1998), *Reinforcement Learning: An introduction*. Cambridge, MA: MIT Press.

U.S. Army. (2008). *Joint Publication 3–0, Joint Operations.*

Valdez, E. (2009) *Analysis of change in population stance on infrastructure using a cultural geography model for stability operations*. Master's thesis, Naval Postgraduate School, Monterey, CA.

Yamauchi, H. (2009). *Cultural geography model (Version 0.7.0)*. Technical report, unpublished, TRAC Monterey, Monterey, CA.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Dr. Chris Darken
   MOVES Institute
   Naval Postgraduate School
   Monterey, California

4. Lieutenant Colonel Jonathan Alt
   TRADOC Analysis Center - Monterey
   Monterey, California

5. Lieutenant Colonel David Hudak
   TRADOC Analysis Center - Monterey
   Monterey, California